Augmenting sstate-cache with ccache

Amir Kirsh

https://bit.ly/YPS-2022_IB_Cache

Yocto Project Summit, 2022-05

About me

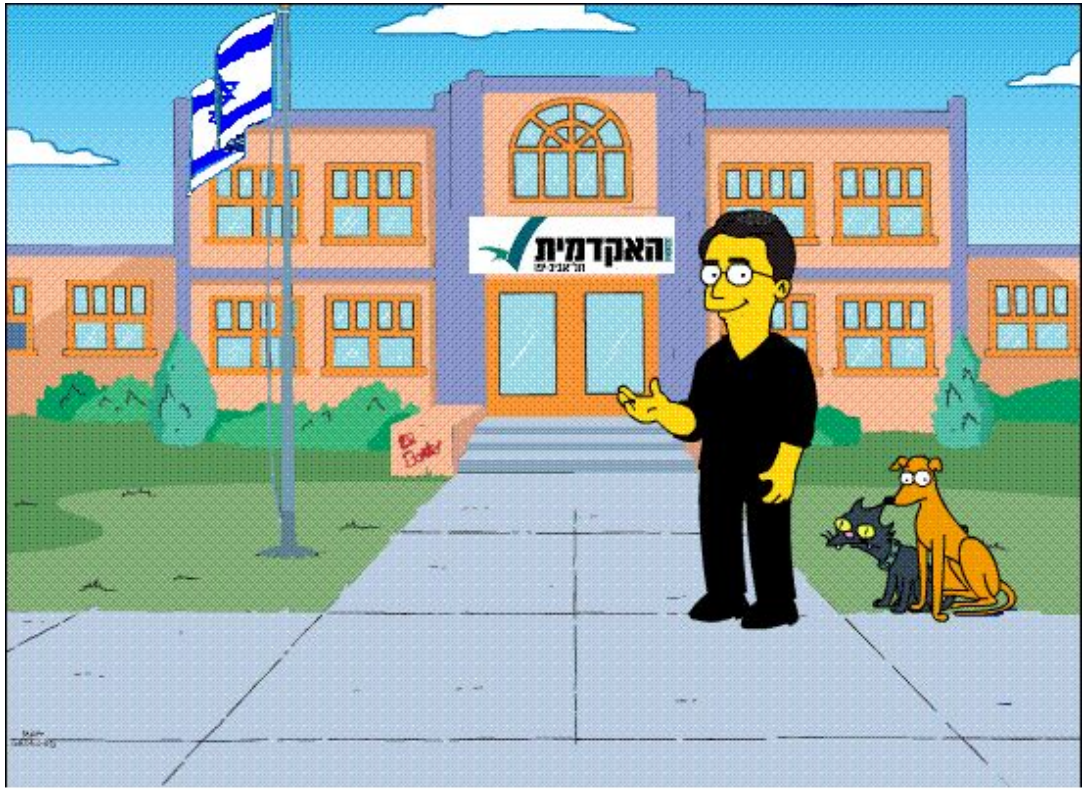
Developer Advocate at

INCREDIBUILD

Lecturer
Academic College of Tel-Aviv-Yaffo
Tel-Aviv University

Member of the Israeli ISO C++ NB

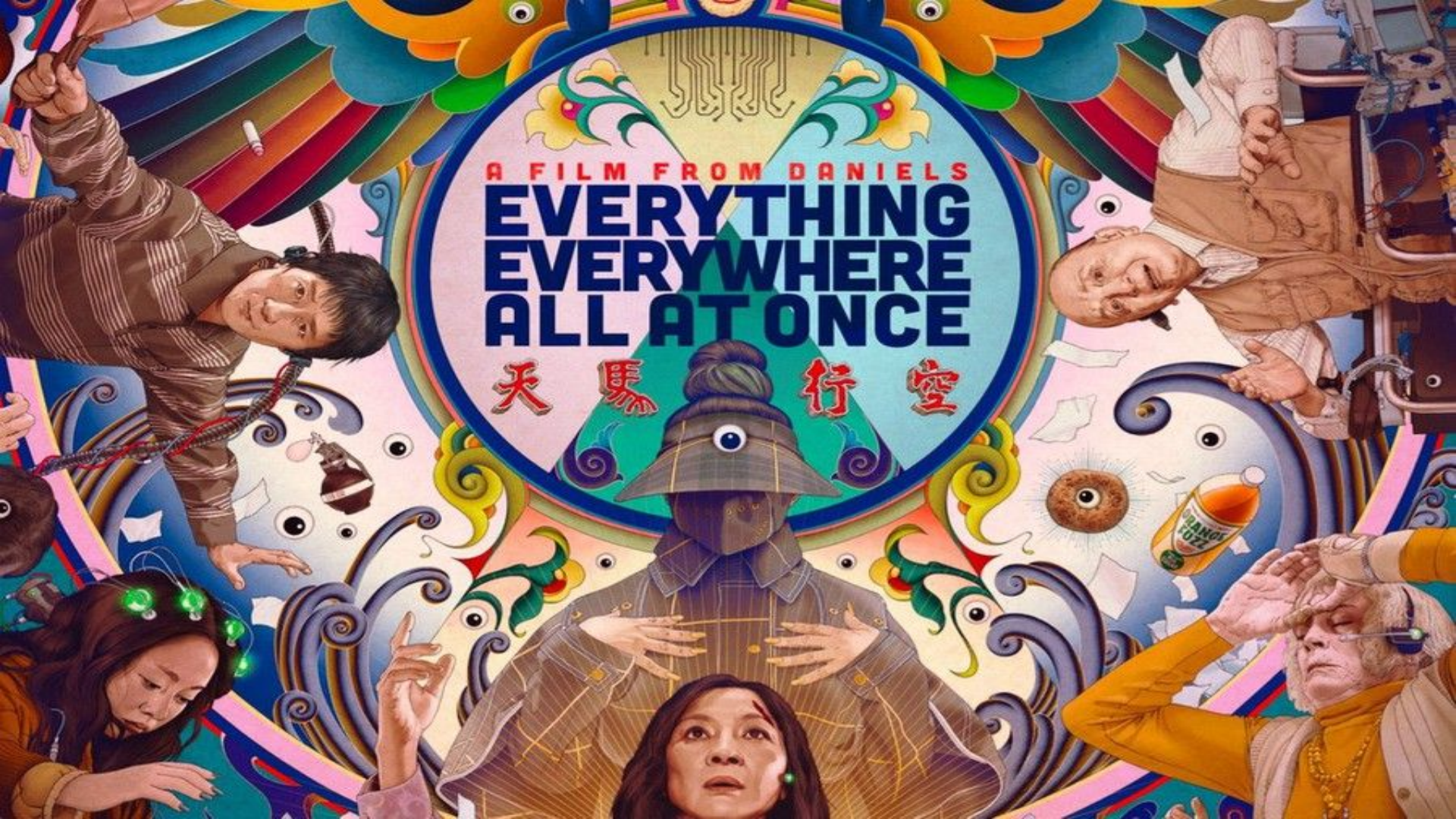
Co-Organizer of the **CoreCpp** conference and meetup group



My talk yesterday:

Breaking down the BitBake build on the process level

https://bit.ly/YPS-2022_IB_bitbake


A FILM FROM DANIELS
EVERYTHING EVERYWHERE ALL AT ONCE
天馬行空

Today's talk:

Augmenting sstate-cache with ccache

https://bit.ly/YPS-2022_IB_Cache

DANIEL AUTEUIL JULIETTE BINOCHE

CACHÉ

(HIDDEN)

★★★★

(Highest Rating)

"Like Hitchcock, only creepier."

-Steven Rea, PHILADELPHIA INQUIRER

WINNER
2005 CANNES FILM FESTIVAL
BEST DIRECTOR

WINNER
2005 EUROPEAN FILM AWARDS
BEST DIRECTOR
BEST ACTOR
BEST FILM

WINNER
BEST FOREIGN FILM
2006 CHICAGO FILM CRITICS ASSOCIATION AWARDS
2006 LOS ANGELES FILM CRITICS ASSOCIATION AWARDS
2006 SAN FRANCISCO FILM CRITICS CIRCLE

You all probably know:

You all probably know:

> *There are only two hard things in Computer Science: cache invalidation and naming things.*
>
> *-- Phil Karlton*

Leon Bambrick
@secretGeek

There are 2 hard problems in computer science: cache invalidation, naming things, and off-by-1 errors.

4:20 PM · Jan 1, 2010

1.6K Reply Share

(Taken from Martin Fowler's page https://martinfowler.com/bliki/TwoHardThings.html)

SSTATE CACHE

- **Several bitbake tasks can use past versions of build artefacts if there have been no changes since the last time you built them**

do_packagedata	Creates package metadata used by the build system to generate the final packages
do_package	Analyzes the content of the holding area and splits it into subsets based on available packages and files
do_package_write_rpm	Creates the actual RPM packages and places them in the Package Feed area
do_populate_lic	Writes license information for the recipe that is collected later when the image is constructed
do_populate_sysroot	Copies a subset of files installed by do_install into the sysroot in order to make them available to other recipes

Slide shamelessly stolen from: http://bit.ly/YPS202105Intro

sstate-cache

- Much more powerful than just a cache for compilations
- Based on a smart checksum mechanism on inputs, to consider whether to use artifacts from the cache*

* See:
https://www.yoctoproject.org/docs/latest/bitbake-user-manual/bitbake-user-manual.html#checksums
https://www.yoctoproject.org/docs/latest/bitbake-user-manual/bitbake-user-manual.html#setscene

sstate hickups (1)

- recipes that do not define correctly their dependencies or outputs, causing either too much work, or worse: things that should get rebuilt do not, ending up in an incorrect output or in build failure
- trying to share sstate-cache between hosts running different Linux distros (even if said to work, may not)

continues next page…

sstate hickups (2)

- **corrupted state when a build is terminated forcefully may affect / terminate next builds***

* During the talk @Richard Purdie explained in the chat: sstate corruption on build failure isn't sstate directly. What usually happens is the TMPDIR sees incomplete corrupted files which break the output on the second build run, which is then placed into sstate in that second run.
(The sstate files are created with an atomic move, so it should be hard to corrupt them directly, it is usually "indirect" corruption).

(you may tackled more / other hickups...)

Let’s make some order in the cache folders

```
$HOME/yocto/build
 |---downloads (DL_DIR)
   Downloaded source cache
 |---sstate-cache (SSTATE_DIR)
   Binary build cache
 |---tmp (TMPDIR)
   All build system’s output
 |---tmp/cache (can be also directly under /build)
   BitBake parser cache (recipes and config files)
```


Check why an artifact was not fetched from sstate

If you wish to compare signatures to see any changes that caused a rebuild, you can use `bitbake-diffsigs` with the `-t` option:

```
$ bitbake-diffsigs -t quilt-native do_configure
```

Alternatively you can compare siginfo files directly:

```
$ bitbake-diffsigs /mnt/sstate/sstate-packagename-checksum1-tgz.siginfo
                   builddir/sstate-cache/sstate-packagename-checksum2-tgz.siginfo
```

https://wiki.yoctoproject.org/wiki/Enable_sstate_cache
https://wiki.yoctoproject.org/wiki/TipsAndTricks/Understanding_what_changed_(diffsigs_etc)

Invalidating the cache

Invalidating the cache (1)

https://www.yoctoproject.org/docs/current/ref-manual/ref-manual.html#ref-tasks-clean

7.2.3. do_clean

Removes all output files for a target from the do_unpack task forward (i.e. do_unpack, do_configure, do_compile, do_install, and do_package).

You can run this task using BitBake as follows:

```
$ bitbake -c clean recipe
```

Running this task does not remove the sstate cache files.

Invalidating the cache (2)

https://www.yoctoproject.org/docs/current/ref-manual/ref-manual.html#ref-tasks-clean

7.2.4. do_cleanall

Removes all output files, shared state (sstate) cache, and downloaded source files for a target [...]

7.2.5. do_cleansstate

Removes all output files and shared state (sstate) cache for a target. [...]

@Peter Kjellerstedt added in comments during the talk: be aware that if you have configured a global sstate cache using SSTATE_MIRRORS, running `bitbake -c cleansstate ...` will only remove the local sstate files. Thus you may still use sstate information from the global cache in the next build.

Invalidating the cache (3)

Controlling the cache on the recipe level:

In the recipe:

```
SSTATE_SKIP_CREATION = "1"
```

or, from outside the recipe (e.g. local.conf):

```
SSTATE_SKIP_CREATION_pn-<recipe_name1> = "1"
SSTATE_SKIP_CREATION_pn-<recipe_name1> = "1"
```

and you can remove sstate for a recipe using:

```
bitbake -c cleansstate <recipe_name>
```

See: How to disable sstate-cache per recipe in Yocto - Stack Overflow
Execute bitbake recipe discarding what sstate_cache is - Stack Overflow

You can also verify if sstate exists for a recipe, using oe-check-sstate, e.g.:

```
oe-check-sstate <yourimage> | grep <recipe_name>
```

To force a recipe to run regardless of what is in the shared-state cache, you can pass the -f | --force option:

```
bitbake -f [other_options...] <recipe_name> ...
```

Invalidating the cache (4)

And you can always just delete the folders...

4.2.7 `build/tmp/`

[...]

As a last resort, to clean up a build and start it from scratch (other than the downloads), you can remove everything in the `tmp` directory or get rid of the directory completely. If you do, you should also completely remove the `build/sstate-cache` directory.

https://docs.yoctoproject.org/ref-manual/structure.html#build-tmp

---

(See also: Why should one delete sstate-cache when deleting tmp in Yocto build? - Stack Overflow).

Invalidating the cache (5)

* This slide originally pointed to a previous version of the manual. @Zach Welch found the new location of this advice, in the current version of the manual, which you now see here.

An advice on forcing invalidation:

3.31.7 Invalidating Shared State to Force a Task to Run

[...]

When you identify an implicit change, you can easily take steps to invalidate the cache and force the tasks to run. The steps you can take are as simple as changing a function's comments in the source code. For example, to invalidate package shared state files, change the comment statements of `do_package` or the comments of one of the functions it calls. Even though the change is purely cosmetic, it causes the checksum to be recalculated and forces the OpenEmbedded build system to run the task again.

https://docs.yoctoproject.org/dev-manual/common-tasks.html#invalidating-shared-state-to-force-a-task-to-run

Invalidating the cache (6)

Are there any cases in which I should just delete the folder:

`build/tmp/cache`

(the BitBake parser cache) – ?

Invalidating the cache (6)

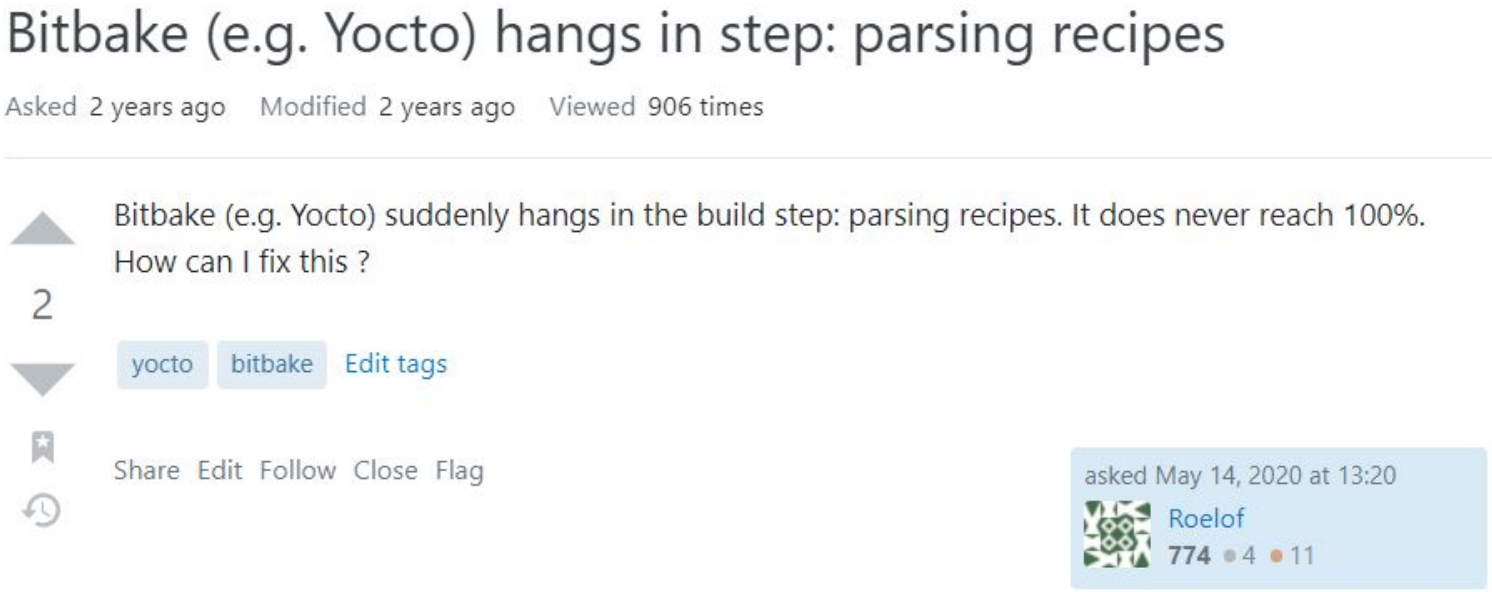
Are there any cases in which I should just delete the folder:

`build/tmp/cache`

(the BitBake parser cache) – ?

Well, maybe...

Invalidating the cache (6)



Bitbake (e.g. Yocto) hangs in step: parsing recipes

Asked 2 years ago Modified 2 years ago Viewed 906 times

Bitbake (e.g. Yocto) suddenly hangs in the build step: parsing recipes. It does never reach 100%.
How can I fix this ?

2

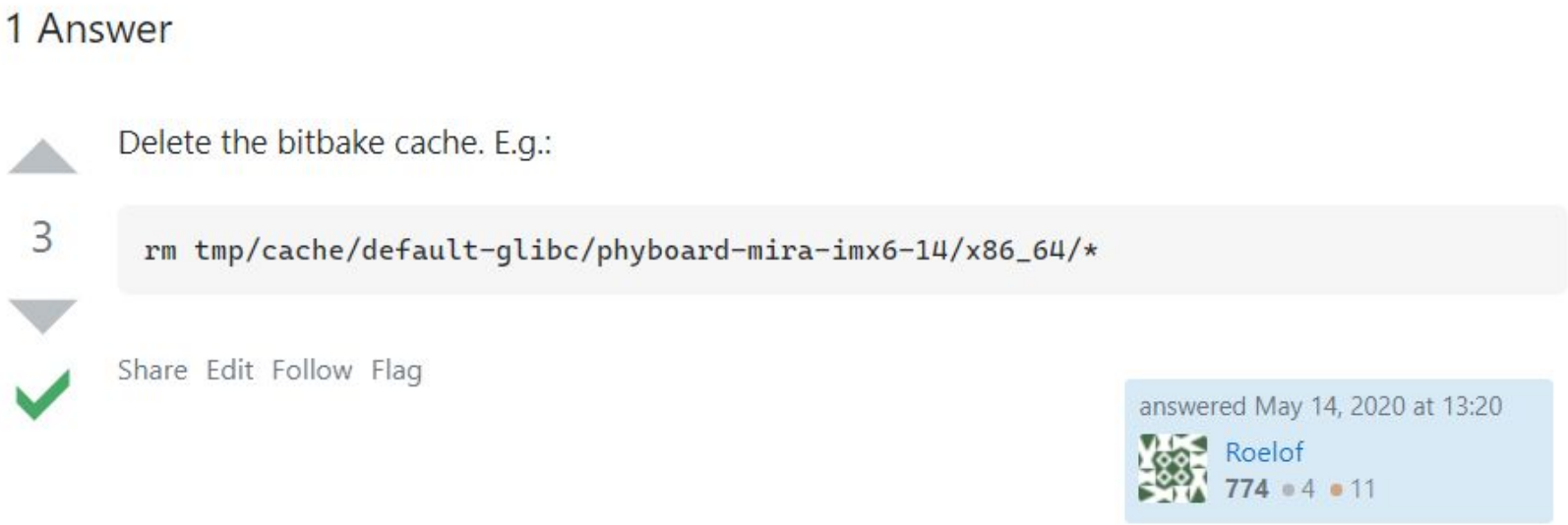
yocto bitbake Edit tags

Share Edit Follow Close Flag

asked May 14, 2020 at 13:20
Roelof
774 ● 4 ● 11

Source: [Bitbake (eg Yocto) hangs in step: parsing recipes - Stack Overflow]

Invalidating the cache (6)



1 Answer

Delete the bitbake cache. E.g.:

3

```
rm tmp/cache/default-glibc/phyboard-mira-imx6-14/x86_64/*
```

Share Edit Follow Flag

answered May 14, 2020 at 13:20

Roelof

774 • 4 • 11

Source: [Bitbake (eg Yocto) hangs in step: parsing recipes - Stack Overflow]

CCache

CCache is an open source caching solution on the C++ compiler level working with g++, clang, MSVC and other compilers.

Only knows how to cache the compilation of a single file (but experts in that!)

You can share the cache between users and on a shared folder or build server.

https://ccache.dev

Why do we need yet another cache??

Why do we need yet another cache? (1)

The sstate-cache requires a rigorous definition of dependencies.
While this is already done for baseline Yocto – we see some of our user base not maintaining these well enough for sstate to work properly.

This may be caused by the user's own code or by inheriting recipes from 3d parties, commercial or open source.

We find users that choose the path of least resistance, throwing bigger machines at the problem and do full rebuilds ("fool rebuilds").

CCache can be an intermediate solution to that problem, by saving compilations.

Why do we need yet another cache? (2)

When a build is aborted by external means (e.g. OOM, a canceled Jenkins job) we saw (and it is documented) that sstate may be saved in an inconsistent state.

In such a case the user needs to clean sstate and to fall back to a full build.

Compiler based caching can then come to assist, since the object files themselves are still available.

We see that ‘do_compile’ composes 50% of the task time (and more).
Letting CCache handle that as a fall back is quite useful.

Our added spice

- **Throwing ccache into the party is useful but might not be too fun, managing configurations and amending recipes to use ccache.***
- **Incredibuild solution configures it all transparently for you, without touching your recipes or config files.**

* during the talk some participants (@Richard Purdie, @Peter Kjellerstedt) suggested in comments that CCache can be added with INHERIT += "ccache" in *local.conf* or *site.conf* - which should make adding CCache even easier. There was also a comment (@Zach Welch) on the need to set the proper size of ccache limit (CCACHE_MAXSIZE).

To Summarize

- Caching is almost crucial when building Yocto.

To Summarize

- Caching is almost crucial when building Yocto.
- Understanding and troubleshooting caching is a required skill to avoid or handle cache related issues.

To Summarize

- Caching is almost crucial when building Yocto.
- Understanding and troubleshooting caching is a required skill to avoid or handle cache related issues.
- Adding ccache as a compiler caching layer can be useful.

To Summarize

- Caching is almost crucial when building Yocto.
- Understanding and troubleshooting caching is a required skill to avoid or handle cache related issues.
- Adding ccache as a compiler caching layer can be useful.
- Using Incredibuild would configure CCache for you, with zero effort 😊 by interception (instead of manual configuration).

+

https://www.incredibuild.com/blog/announcing-incredibuild-support-for-yocto
https://www.incredibuild.com/lp/yocto

DANIEL AUTEUIL JULIETTE BINOCHE

CACHÉ

(HIDDEN)

★★★★

(Highest Rating)

"Like Hitchcock, only creepier."

-Steven Rea, PHILADELPHIA INQUIRER

WINNER
2005 CANNES FILM FESTIVAL
BEST DIRECTOR

WINNER
2005 EUROPEAN FILM AWARDS
BEST DIRECTOR
BEST ACTOR
BEST FILM

WINNER
BEST FOREIGN FILM
2006 CHICAGO FILM CRITICS ASSOCIATION AWARDS
2006 LOS ANGELES FILM CRITICS ASSOCIATION AWARDS
2006 SAN FRANCISCO FILM CRITICS CIRCLE

Augmenting sstate-cache with ccache

https://bit.ly/YPS-2022_IB_Cache

amir.kirsh@incredibuild.com





yocto PROJECT | THE LINUX FOUNDATION